



Gestão de Configuração de Software

- Conceitos Principais -

ES06 – Especialização em Engenharia de Software

Prof.: Misael Santos (misael@gmail.com)

Ago/2010

Agenda

- Configuração
- Itens de Configuração
- Versionamento
- Repositório
- Espaço de Trabalho (*Workspace*)
- Baselines
- Build
- Reproducibilidade
- CCB

Configuração

- Os itens que compreendem toda a informação produzida como parte do processo de software são chamados coletivamente de *configuração de software*.

Configuração

- É a designação geral para o conjunto de *itens de configuração** de um projeto de software.

Itens de Configuração de Software

- É a informação criada como parte do processo de engenharia de software. (SCI, Pressman, 2006)
- É a designação geral de qualquer artefato ou produto de software mantido sob gestão de configuração/mudança.
- São itens de informação selecionados.

Itens de Configuração de Software

- Em geral é:
 - Um produto de software ou
 - Um produto de desenvolvimento de software
- Deve possuir uma identificação única
 - Regras de Nomenclatura devem ser utilizadas

Itens de Configuração de Software

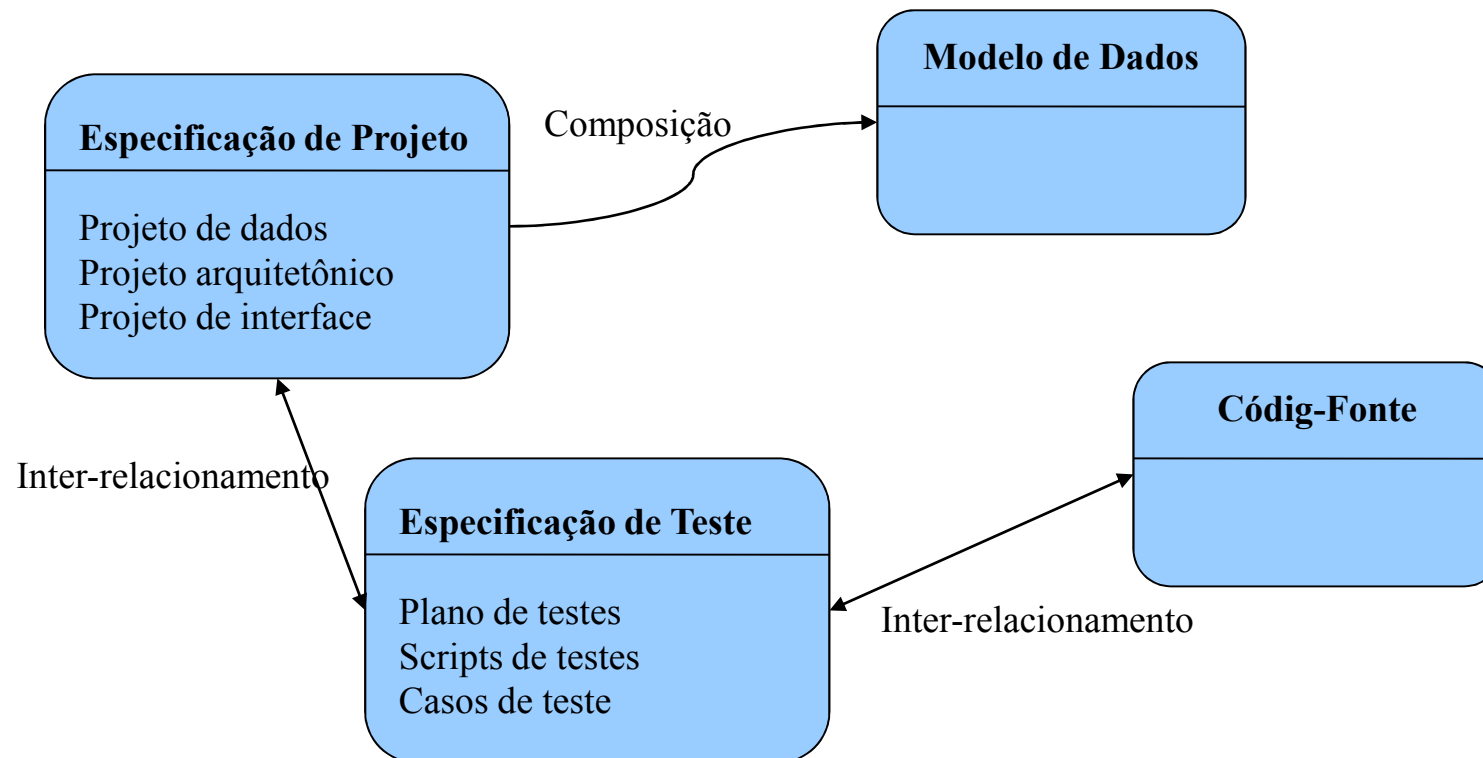
- Exemplos:
 - um plano de projeto
 - um cronograma
 - uma especificação de caso de uso
 - um modelo ou parte de um modelo
 - **código-fonte ***
 - um módulo executável ou componente,
 - um arquivo de help
 - um script de teste

Itens de Configuração de Software

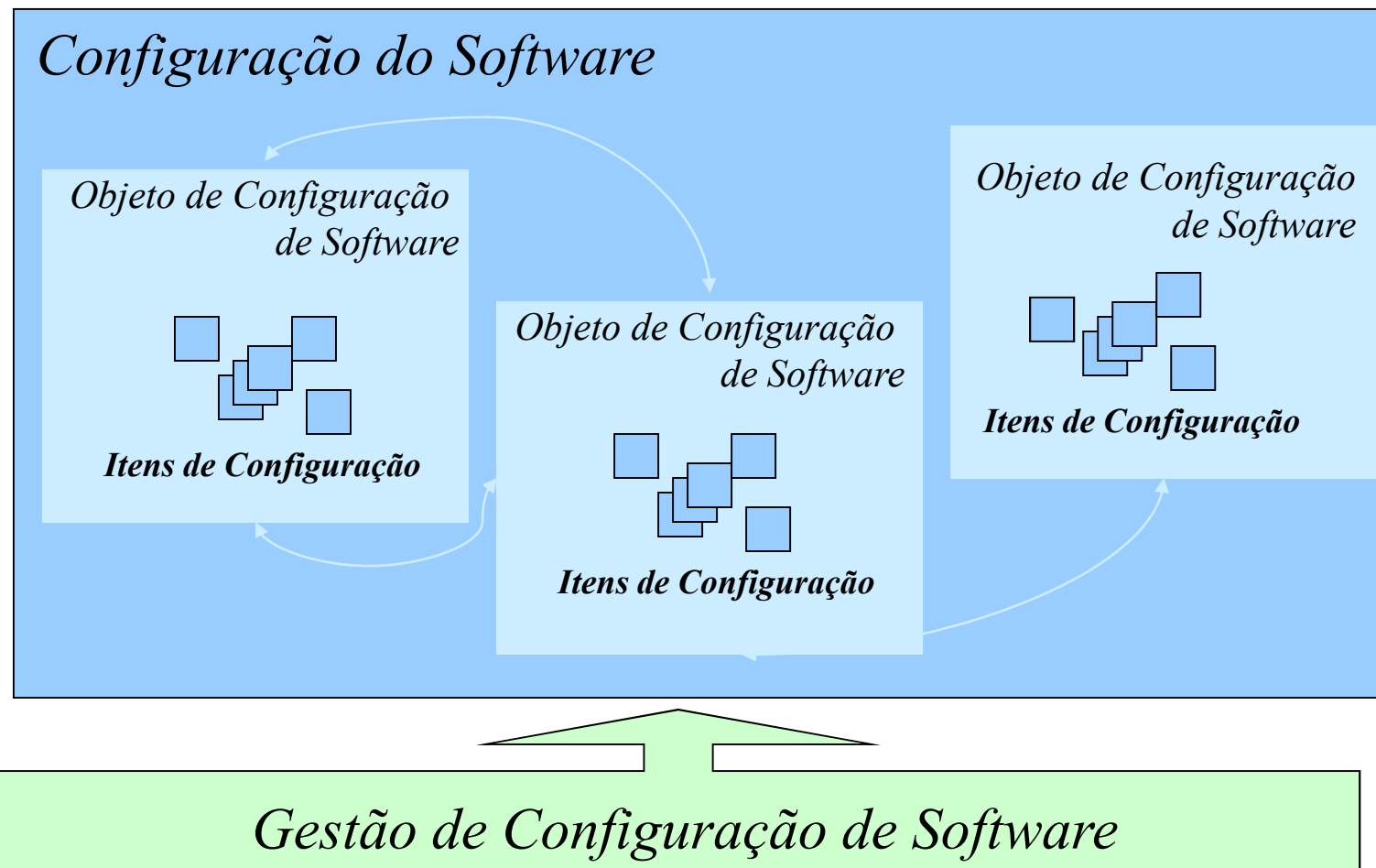
- Muitas organizações colocam também ferramentas de software sob controle de configuração. Ex:
 - Versões específicas de editores
 - Compiladores
 - Navegadores
 - Outras ferramentas...
- Importante para a *Reproducibilidade**

Itens de Configuração de Software

- Itens de Configuração podem ser agrupados em *Objetos de Configuração*.



Itens de Configuração de Software



Versionamento

- Versionamento é a disciplina através da qual são preservadas as versões de um artefato, de modo sistemático e seguro, não limitado em número de versões.
- A cada alteração consolidada no repositório uma nova versão do item deve ser gerada.
- Todas as versões devem ser armazenadas e identificadas.
- Revisões = versões individuais de cada item.

Versionamento

- Versão:
 - ⊕ Estado definido de um objeto num dado momento.
 - ⊕ “Fotografia” do objeto.

Versionamento

- Tipos de versionamento:
 - *Delta Negativo* - armazena-se integralmente a versão mais recente e as diferenças (deltas) existentes até então. (**Mais Implementado**)
 - *Delta Positivo* - armazena-se a versão mais antiga e, para montar as versões mais recentes, processam-se as diferenças (deltas) armazenadas.

Versionamento

>_

arquivo1.c

1
2
3
4
5
6

document2.doc

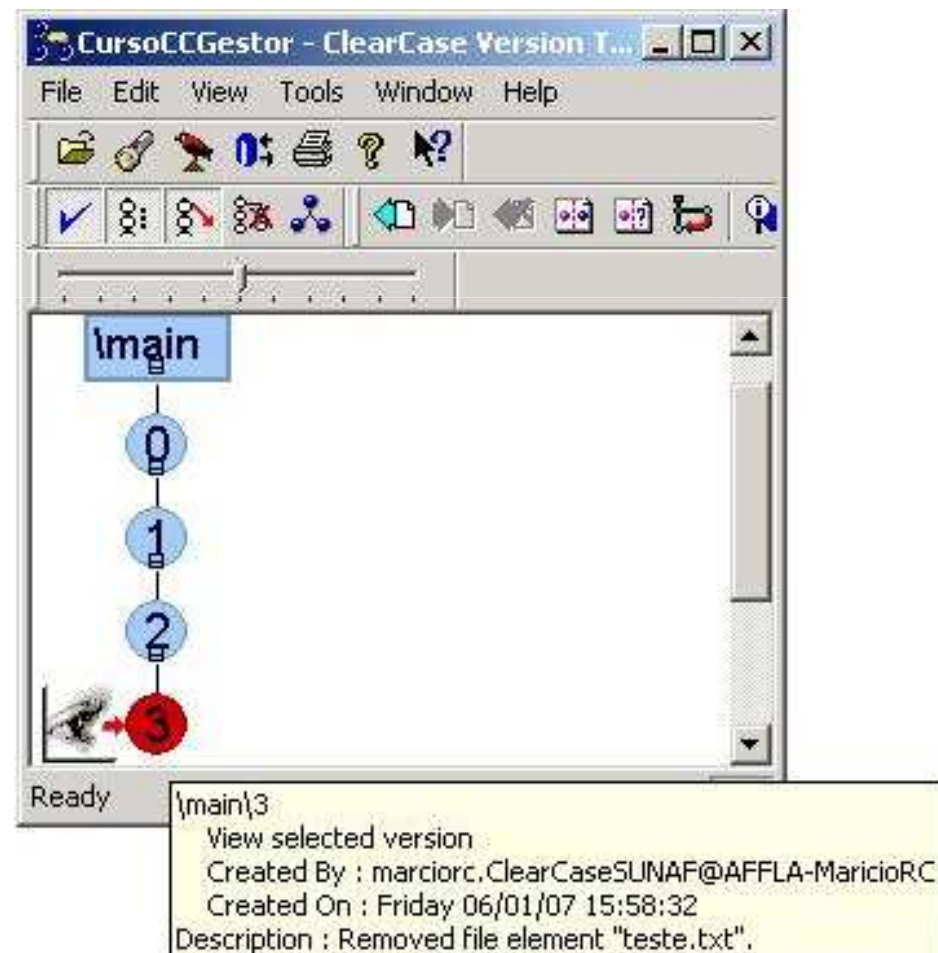
1
2

classeA.java

1
2
3
4

Versionamento

- Árvore de versões do ClearCase



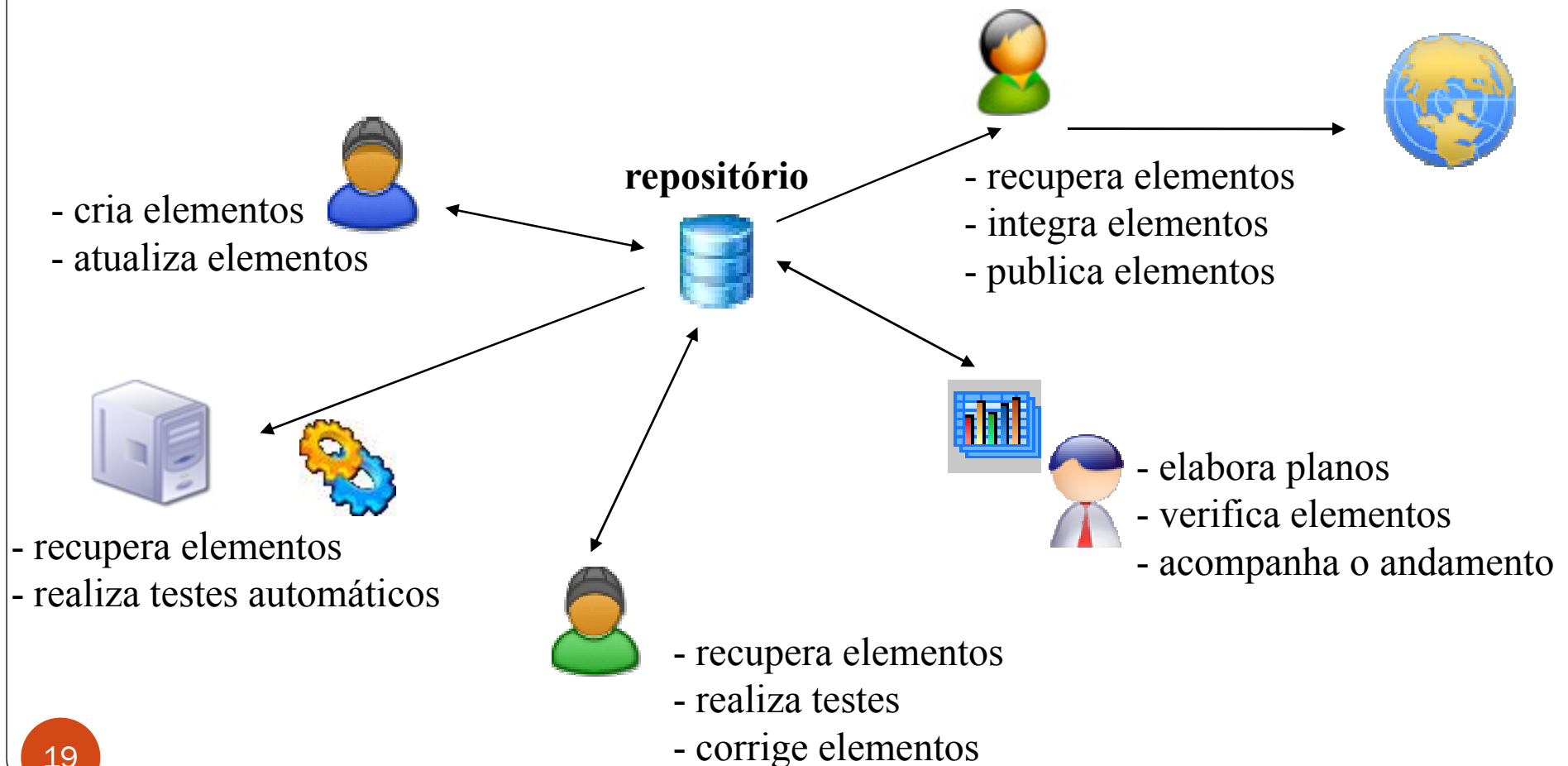
Versionamento

- Deve ser possível recuperar versões anteriores de um item.
- Além de artefatos, podemos versionar configurações completas ou parciais de um software
- É, portanto, aceitável falar-se em:
 - ✦ Versão de um artefato (documento, cronograma, modelo, programa, etc)
 - ✦ Versão de um sistema

Repositório



Repositório



Repositório

- É o conjunto de mecanismos e estruturas de dados que permite a uma equipe de software gerir modificação de modo efetivo.
- É um local sob controle de acesso onde são armazenados os itens de configuração.

Repositório

- Deve permitir:
 - Controle de acesso
 - O versionamento dos elementos
 - Representar marcos de projeto ou versões de produção específicas
 - Pistas de auditoria das modificações
 - O que, quando, quem e por que.

Repositório

- Existem diversas ferramentas de mercado:
 - Rational ClearCase
 - CVS (*Concurrent Versions System*)
 - Subversion
 - Git
 - Mercurial

Espaço de Trabalho (*Workspace*)



Espaço de Trabalho (*Workspace*)

- Um lugar onde o desenvolvedor possa trabalhar isoladamente sobre os seus artefatos enquanto ele finaliza uma tarefa sem interferências externas.
- Utilizado para:
 - Criação/edição de artefatos
 - Operações de gerenciamento
 - Consultas

Espaço de Trabalho (*Workspace*)

- **Visões de Imagem** (ou Visão Estática ou Snapshots):
 - Apresenta ao desenvolvedor um ambiente de trabalho estável e isento de mudanças.
 - Quando um desenvolvedor deseja “enxergar” mudanças realizadas por outros grupos, ele pode atualizar sua visão.
 - Este formato de trabalho é conhecido por “modelo pull”, orientando sempre pela busca da informação desejada, ao invés de empurrar as mudanças.
- **Visões Dinâmicas:**
 - Não armazenam nada localmente, dependem de atualizações imediatas realizadas através da rede. São recomendadas nas seguintes situações:
 - Espaço restrito no ambiente do desenvolvedor
 - Necessidade de compartilhamento de certos objetos
 - Membros de grupos virtuais necessitam trabalhar com as últimas versões de um código

Baselines

- Também chamadas de:
 - Referenciais (*Pressman*)
 - Linhas de Referências
 - Linhas de Base
 - Linhas-base

Baselines

- “*Uma especificação ou produto que foi formalmente revisto e aprovado, o qual daí em diante serve como base para o desenvolvimento futuro e que pode ser modificado apenas por meio de procedimentos formais de controle de modificação.” (IEEE Std nº. 610.12-1990)*

Baselines

- Uma baseline é uma “fotografia” de um conjunto de itens de configuração em um determinado momento do processo de desenvolvimento.
- A configuração do software em um ponto discreto no tempo.
- Normalmente é gerada ao final de uma fase do desenvolvimento ou demandada pelo líder do projeto.

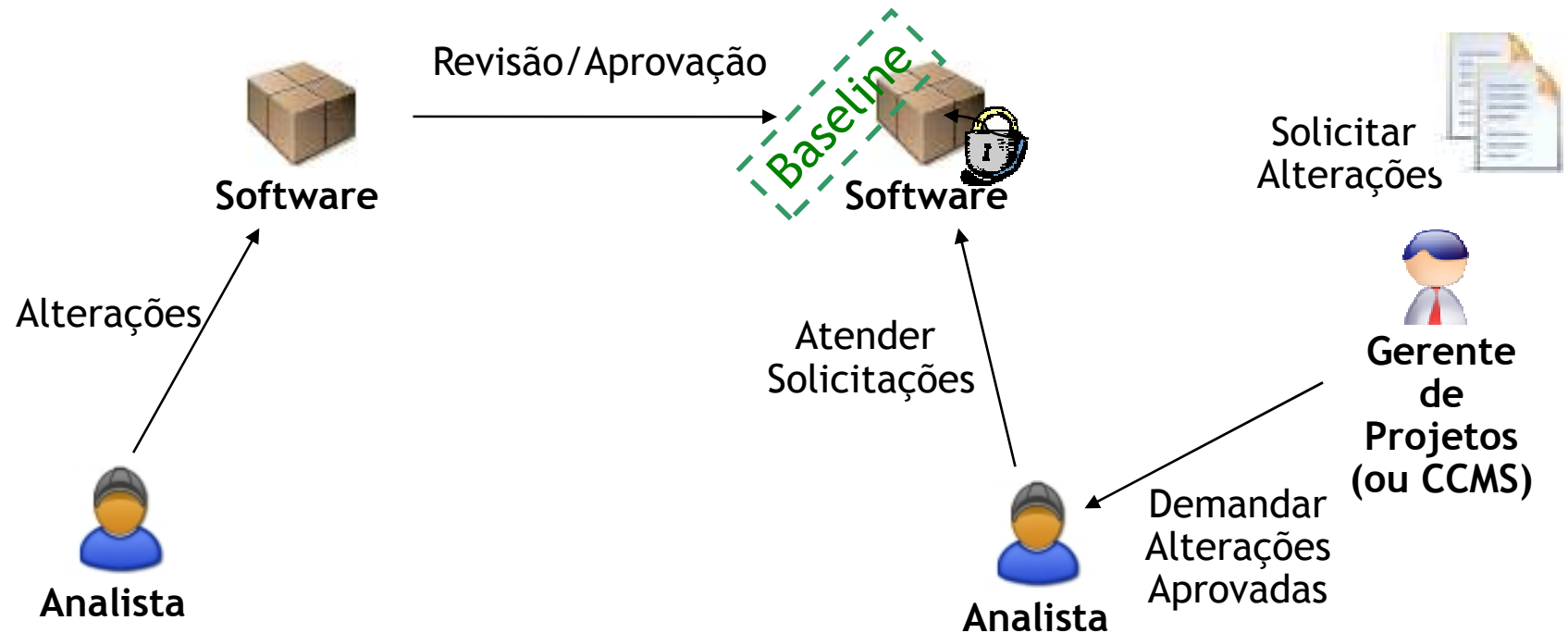
Baselines

- Representam Marcos no Projeto (*Milestones*), ex:
 - Final de especificação dos requisitos
 - Modelos documentados e revisados
 - Final da implementação de um módulo (incremento)
 - Erros encontrados e corrigidos (Testes, Homologação)
 - Implantação de uma Versão do Sistema

Baselines

- Origem da versão de todos os ambientes:
 - Testes
 - Homologação
 - Produção

Baselines

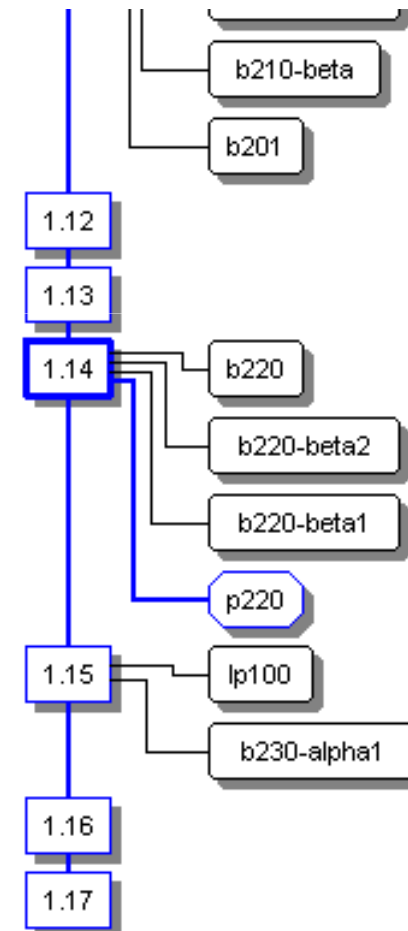


Baselines

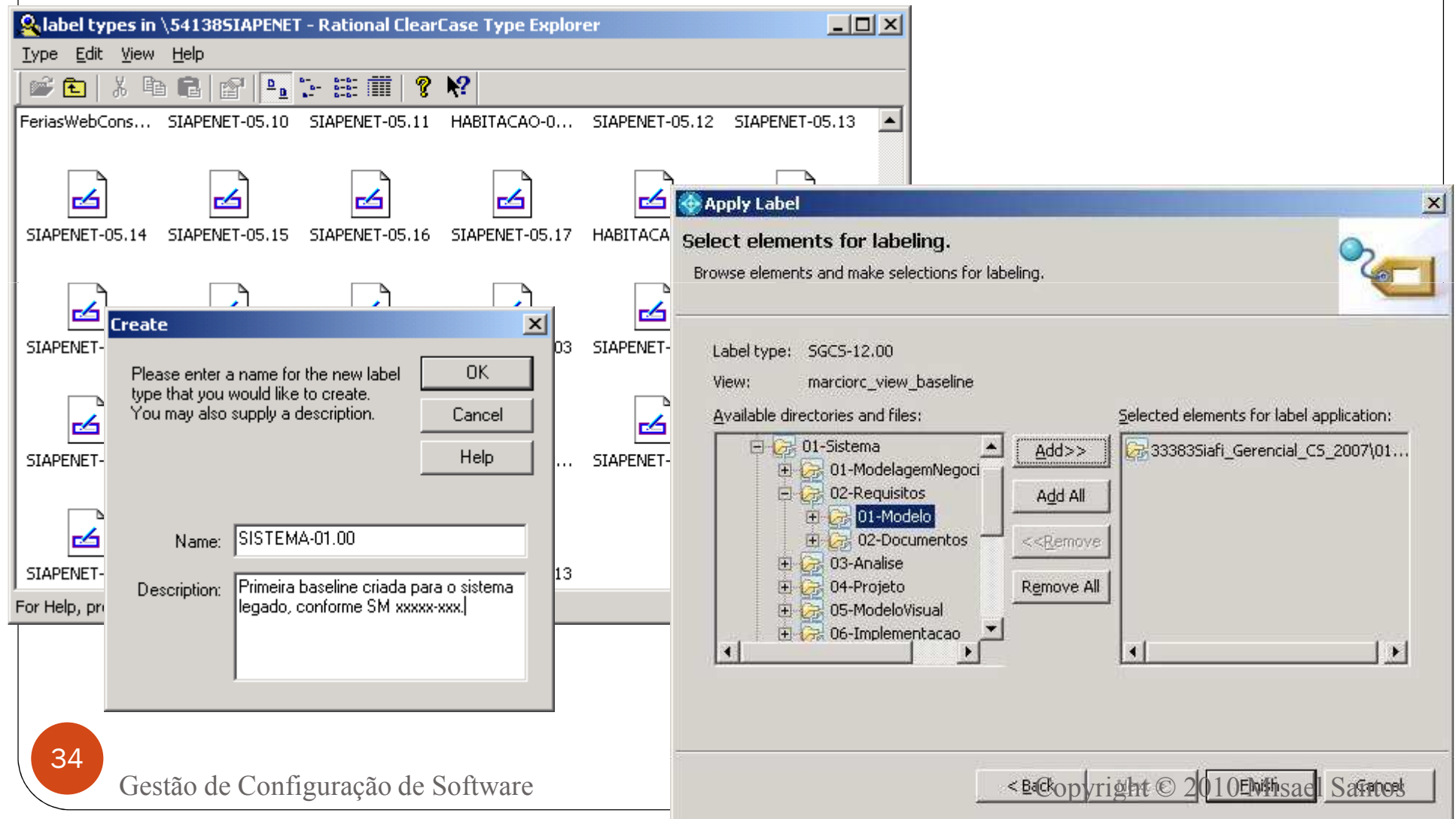
- Ferramentas possuem recursos próprios para a administração de baselines
 - CVS, Subversion
 - tag
 - ClearCase
 - Apply Label

Baselines no CVS

```
bash-2.05a$ cvs tag pre_alpha_0-1
cvs server: Tagging .
T Changelog
T INSTALL
T Makefile
T README
T TODO
cvs server: Tagging doc
cvs server: Tagging doc/design
T doc/design/AcceptanceTest.doc
T doc/design/Analysis.rtf
T doc/design/Requirements.doc
T doc/design/Specification.rtf
cvs server: Tagging doc/plan
T doc/plan/Schedule.rtf
cvs server: Tagging lib
cvs server: Tagging man cvs server:
Tagging src
T src/config.h
T src/main.c
```



Baselines no ClearCase



Build

- Representa uma versão que pode ser ainda incompleta do sistema em desenvolvimento, mas com certa estabilidade
- Costuma apresentar limitações conhecidas
- Pode incluir código-fonte, documentação, arquivos de configuração, base de dados, etc
- A política de geração das builds deve ser bem definida na estruturação do ambiente

Build

- Durante o projeto podem ser criadas várias *builds*
 - ✚ A frequência varia de acordo com a necessidade do projeto
- A *build* é o resultado principal de uma integração
- Pode ser gerada automaticamente com a utilização de ferramentas

Integração

- Atividade que consiste em “juntar” partes do software conforme vão ficando prontas
 - ✦ Participam da atividade de integração itens já testados isoladamente (teste de unidade)
 - ✦ A integração pode incluir sincronização entre modelo e código
 - ✦ O produto principal da integração é uma *build*

Reproducibilidade

- A habilidade de reproduzir/gerar uma versão anterior do sistema.
- A partir do conteúdo de uma baseline conseguir obter uma versão estável do produto.

Reprodutibilidade

- ***The Baseline Reproducibility Principle:***
Uma baseline deve ser reproduzível. Devemos ser capazes de reproduzir a “configuração” e o conteúdo de todos os elementos necessários para reproduzir uma versão “liberada” (*released*) do produto.
(Brad Appleton)

CCB

- *Configuration Control Board* – Comitê de Controle da Configuração
- Ou ainda CMCS (Comitê de Mudanças na Configuração do Software)



CCB

- Têm autoridade para aprovar ou rejeitar solicitações de alterações.

CCB

- Responsabilidades:
 - Analisar as solicitações de mudança
 - Controlar o escopo do projeto
 - Reuniões com frequência adequada ao ritmo das solicitações de mudança
 - Envolver *stakeholders* no processo de priorização e decisão
 - Balanço entre o nível de controle desejado e *overhead* suportado
 - Questões menores devem ser resolvidas pelo líder do projeto junto à equipe, reduzindo o *overhead* do CCB

Características do CCB

- Composição multidisciplinar
 - SQA, gerente, cliente, arquiteto
- Profissionais com grande capacidade de comunicação e negociação
- Pode apresentar uma estrutura hierarquica dependendo do tamanho e da quantidade de *stakeholders* e sistemas envolvidos
- Em pequenos projetos pode ser representando unicamente pelo Lider de Projeto ou outro profissional designado.

Referências

- PRESSMAN, R. S., *Engenharia de Software*, 6ª. ed., 2006.
- HASS A. M. J., *Configuration Management Principles and Practice*, Addison Wesley, 432p, 2002.
- FREDERICKST. *Software Configuration and Integration Management*, Marquette University, 2001.
- *PSDS – Processo SERPRO de Desenvolvimento de Soluções*, SERPRO, 2006.
- Brad Appleton's ACME Blog, disponível em:
<http://bradapp.blogspot.com>